Adversarial Feature Matching for Text Generation

**Presenter:** Yizhe Zhang

**Joint work with:** Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, Lawrence Carin

Duke University

August 9, 2017
A game between:
- Discriminative model D
- Generative model G

G: trained to maximize the probability of D making a mistake

D: trained to estimate the probability that a sample came from data distribution rather than G
**Motivation & contributions**

- **Motivation:** Generate realistic-looking text via adversarial training.

- **Difficulties:** (due to discrete nature of text)
  - Synthetic data is not directly differentiable.
  - Transitions in text are less smooth than in images. → mode collapsing.

- **Our approach:**
  - Discretization approximations using *Gumbel-softmax*.
  - Ameliorating mode-collapsing issue via *feature moment matching*. 
We specify an LSTM generator to translate a latent code vector, $z$, into a synthetic sentence $\tilde{s}$.

All other words in the sentence are sequentially generated using the RNN, based on previously generated words, until the end-sentence symbol is generated.
argmax operation is not differentiable.

We consider a **Gumbel-softmax** approach to approximate argmax operation.

\[ y_{t-1} = W_e \text{softmax}(Vh_{t-1} \odot 1/\tau). \] (1)

where \( \odot \) represents the element-wise product. \( W_e \in \mathbb{R}^{k \times V} \) is a word embedding matrix. \( V \) is a weight matrix. Note that when \( \tau \to 0 \), this approximation approaches argmax operation.
CNNs weight each word equally and are empirically better at abstracting features particularly with long sentences.

A sentence is represented as a matrix $\mathbf{X} \in \mathbb{R}^{k \times T}$, followed by a convolution operation.

A max-over-time pooling operation is then applied.

**Figure:** CNN discriminator
The adversarial game is the following:
- $D(\cdot)$ attempts to select informative sentence features.
- $G(\cdot)$ aims to match these features.
- Features are selected according to \textit{syn/real discrimination ability}, \textit{latent code reconstruction} and \textit{moment matching precision}.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{model_scheme.png}
\caption{Model scheme of TextGAN.}
\end{figure}
Feature moment matching (for G)

- Optimization schemes:

\[ \mathcal{L}_G = \mathcal{L}_{MMD^2} \]
\[ \mathcal{L}_D = \mathcal{L}_{GAN} + \lambda_r \mathcal{L}_{recon} - \lambda_m \mathcal{L}_{MMD^2} \]

- For G, consider a moment matching loss over feature vector using maximum mean discrepancy (MMD).

\[ \mathcal{L}_{MMD^2} = \| \mathbb{E}_{x \sim X} \phi(x) - \mathbb{E}_{y \sim Y} \phi(y) \|_{\mathcal{H}}^2 \]
\[ = \mathbb{E}_{x \sim X} \mathbb{E}_{x' \sim X} [k(x, x')] \]
\[ + \mathbb{E}_{y \sim Y} \mathbb{E}_{y' \sim Y} [k(y, y')] - 2 \mathbb{E}_{x \sim X} \mathbb{E}_{y \sim Y} [k(x, y)]. \tag{2} \]

- With a Gaussian kernel, minimizing the MMD objective \(\Leftrightarrow\) minimizing all order of moments of two empirical distributions.
Feature moment matching (for G)

- Vanilla GAN: D independently judge each syn/real data.
- The **MMD loss** for G: match distributions, enforce diversity.
- The gradient signal back-propagated from feature layer is more direct.
Feature moment matching (for D)

- Optimization schemes:

\[
\mathcal{L}_G = \mathcal{L}_{MMD^2} \\
\mathcal{L}_D = \mathcal{L}_{GAN} + \lambda_r \mathcal{L}_{recon} - \lambda_m \mathcal{L}_{MMD^2} \\
\mathcal{L}_{GAN} = -\mathbb{E}_{s \sim S} \log D(s) - \mathbb{E}_{z \sim p_z} \log[1 - D(G(z))] \\
\mathcal{L}_{recon} = ||\hat{z} - z||^2,
\]

- The reconstruction loss in D : select the most representative (information-preserving) features.

- The MMD loss in D: select the most challenging features.
Pre-training strategy

- For G, pretrained by using sequence-to-sequence language model.
- For D, *permutation training* strategy: randomly swap two words to construct a *tweaked* sentence counterpart.

![Permutation training diagram](image)

**Figure:** Permutation training
Variants

- Problems: a minibatch of data points is not densely sampled in feature space with high dimension (900).
- Variants:
  - **(MMD-L)**: Mapping feature space to lower dimension (by $D$).
  - **(MM)**: Use *accumulated batches*, match *first-order* moment.

\[
L_G = (\mu_s - \mu_r)^T(\mu_s - \mu_r)
\]

- **(CM)**: Use accumulated batches, match *first-order* and *second-order* moment, which can be interpreted as an lower-bound of JSD between two MVNs:

\[
L_G = \text{tr}(\Sigma_s^{-1}\Sigma_r + \Sigma_r^{-1}\Sigma_s) + (\mu_s - \mu_r)^T(\Sigma_s^{-1} + \Sigma_r^{-1})(\mu_s - \mu_r)
\]

\(\Sigma_{(s/r)}\) and \(\mu_{(s/r)}\) are (accumulated) covariance matrix and mean vector for syn/real feature vector.
Empirical evaluation

- **Dataset:** 0.5M Arxiv sentences + 0.5M BookCorpus sentences.
- **Evaluation:** Kernel density estimation (KDE).
- **Evaluation:** Corpus-level BLEU score.
- Compared with baseline auto-encoder, variational auto-encoder and seqGAN [Yu et. al. 2016]
Produce novel phrases by imagining concept combinations. (b)

In general, the synthetic sentences seem syntactically reasonable.

However, the semantic meaning is less well preserved with long sentences. (e)

Table: Sentences generated by textGAN.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>we show the joint likelihood estimator (in a large number of estimating variables embedded on the subspace learning) .</td>
</tr>
<tr>
<td>b</td>
<td>this problem achieves less interesting choices of convergence guarantees on turing machine learning .</td>
</tr>
<tr>
<td>c</td>
<td>in hidden markov relational spaces , the random walk feature decomposition is unique generalized parametric mappings.</td>
</tr>
<tr>
<td>d</td>
<td>i see those primitives specifying a deterministic probabilistic machine learning algorithm .</td>
</tr>
<tr>
<td>e</td>
<td>i wanted in alone in a gene expression dataset which do n’t form phantom action values .</td>
</tr>
<tr>
<td>f</td>
<td>as opposite to a set of fuzzy modelling algorithm , pruning is performed using a template representing network structures .</td>
</tr>
</tbody>
</table>
Experimental Result

Moment Matching

Figure: Moment matching comparison. Left: expectations of latent features from real vs. synthetic data. Right: elements of covariance matrix for real and synthetic data, respectively.
**Table:** Intermediate sentences produced from linear transition between two points.

<table>
<thead>
<tr>
<th>textGAN</th>
<th>AE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
<td></td>
</tr>
<tr>
<td>our methods apply novel approaches to solve modeling tasks.</td>
<td>our methods apply to train UNK models involving complex.</td>
</tr>
<tr>
<td>- our methods apply novel approaches to solve modeling.</td>
<td></td>
</tr>
<tr>
<td>- our methods apply two different approaches to solve computing.</td>
<td>our methods solve use to train UNK.</td>
</tr>
<tr>
<td>- our methods achieve some different approaches to solve computing.</td>
<td>our approach show UNK to models exist.</td>
</tr>
<tr>
<td>- our methods achieve the best expert structure detection.</td>
<td>that supervised algorithms show to UNK speed.</td>
</tr>
<tr>
<td>- the methods have been different related tasks.</td>
<td>that address algorithms to handle UNK.</td>
</tr>
<tr>
<td>- the guy is the minimum of UNK.</td>
<td>that address versions to be used in.</td>
</tr>
<tr>
<td>- the guy is n't easy tonight.</td>
<td>i believe the means of this attempt to cope.</td>
</tr>
<tr>
<td>- i believe the guy is n't smart okay?</td>
<td>i believe it 's we be used to get.</td>
</tr>
<tr>
<td>- i believe the guy is n't smart.</td>
<td>i believe it i 'm a way to belong.</td>
</tr>
<tr>
<td><strong>B</strong></td>
<td></td>
</tr>
<tr>
<td>i believe i 'm going to get out.</td>
<td></td>
</tr>
</tbody>
</table>
Higher BLEU, lower KDE is better.

Table: Quantitative results using BLEU-2,3,4 and KDE.

<table>
<thead>
<tr>
<th>Method</th>
<th>BLEU-4 ± 0.01</th>
<th>BLEU-3 ± 0.02</th>
<th>BLEU-2 ± 0.02</th>
<th>KDE(nats) ± 0.02</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE</td>
<td>0.01 ± 0.01</td>
<td>0.11 ± 0.02</td>
<td>0.39 ± 0.02</td>
<td>2727 ± 42</td>
</tr>
<tr>
<td>VAE</td>
<td>0.12 ± 0.06</td>
<td>0.40 ± 0.06</td>
<td>0.61 ± 0.07</td>
<td>2025 ± 25</td>
</tr>
<tr>
<td>seqGAN</td>
<td>0.04 ± 0.04</td>
<td>0.30 ± 0.08</td>
<td>0.67 ± 0.04</td>
<td>2019 ± 53</td>
</tr>
<tr>
<td>textGAN(MM)</td>
<td>0.09 ± 0.04</td>
<td>0.42 ± 0.04</td>
<td>0.77 ± 0.03</td>
<td>1823 ± 50</td>
</tr>
<tr>
<td>textGAN(CM)</td>
<td>0.12 ± 0.03</td>
<td>0.49 ± 0.06</td>
<td>0.84 ± 0.02</td>
<td>1686 ± 41</td>
</tr>
<tr>
<td>textGAN(MMD)</td>
<td>0.13 ± 0.05</td>
<td>0.49 ± 0.06</td>
<td>0.83 ± 0.04</td>
<td>1688 ± 38</td>
</tr>
<tr>
<td>textGAN(MMD-L)</td>
<td>0.11 ± 0.05</td>
<td>0.52 ± 0.07</td>
<td>0.85 ± 0.04</td>
<td>1684 ± 44</td>
</tr>
</tbody>
</table>
Conclusion

- We introduced a novel approach for text generation using adversarial training.
- We discussed several techniques to alleviate practical issues when training GAN on text domain.
- We demonstrated that the proposed model delivers superior performance compared to related approaches.
Q&A

dpaper:  https://arxiv.org/abs/1706.03850

code:  https://github.com/dreasy_snail/textGAN_public

poster:  #89 Wednesday
CNNs weight each word equally and are empirically better at abstracting features particularly with long sentences.

A sentence is represented as a matrix $X \in \mathbb{R}^{k \times T}$, by concatenating its word embeddings as columns.

A convolution operation involves a filter $W_c \in \mathbb{R}^{k \times h}$, applied to a window of $h$ words to produce a new feature.

A max-over-time pooling operation is then applied to the feature map.

**Figure:** CNN discriminator